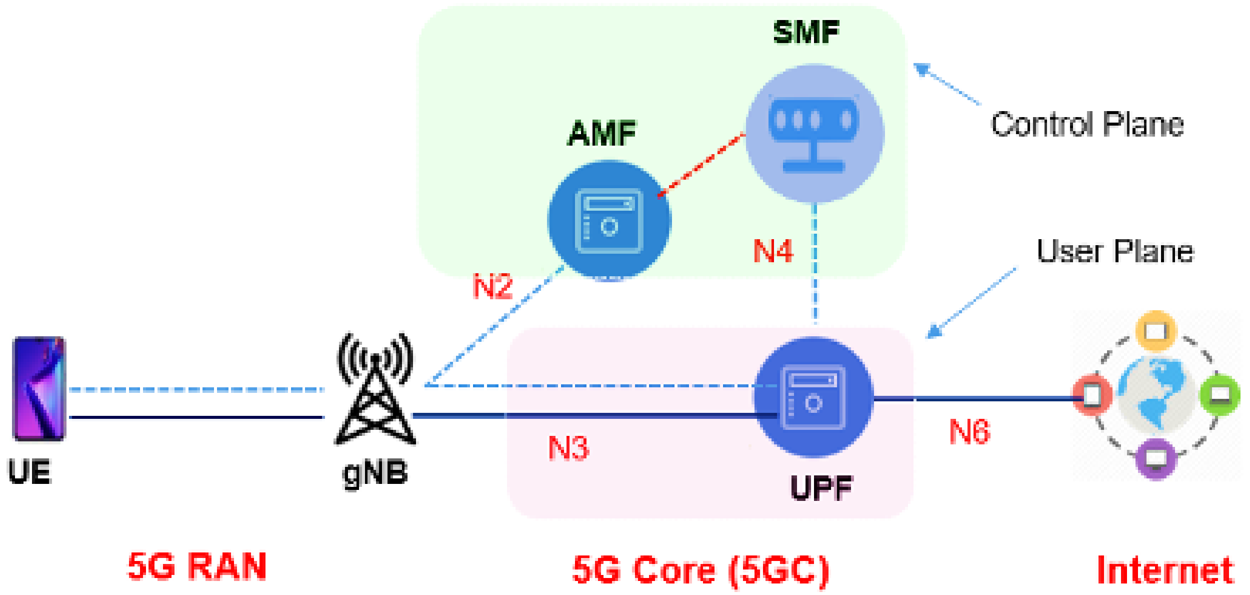
Wireless Networks

5G Wireless Network (Lab)

Emulate a 5G network deployment in comnetsemu.
Demonstrate distributed UPF deployment and slice-base UPF selection

Hassan Shabir
LM CS and Networking Student
h.shabir@studenti.unipi.it

5G Architecture diagram



OPEN SOURCE 5G CORE IMPLEMETATIONS

1. OpenAirInterface:
   1. GitHub repository: https://github.com/openairinterface
   2. Documentation: https://openairinterface.org/oai-5g-core-network-project/
2. OPEN5gs
   1. GitHub repository: https://github.com/open5gs/open5gs
   2. Documentation: https://open5gs.org/

OPEN SOURCE RAN IMPLEMETATIONS

1. OpenAirInterface:
   1. GitHub repository: https://github.com/openairinterface
   2. Documentation: https://openairinterface.org/oai-5g-ran-project/

2. UERAMSIM
   1. GitHub repository https://github.com/aligungr/UERANSIM

OPEN5GS

Open5GS is a C-language Open Source implementation of 5GC and EPC, i.e. the core network of NR/LTE network.

Open5GS supports both Standalone (SA) and Non-Standalone (NSA) deployment options for 5G networks. SA deployment mode requires a complete 5G core network implementation, while NSA deployment mode uses the existing 4G core network to provide certain service

Open5GS

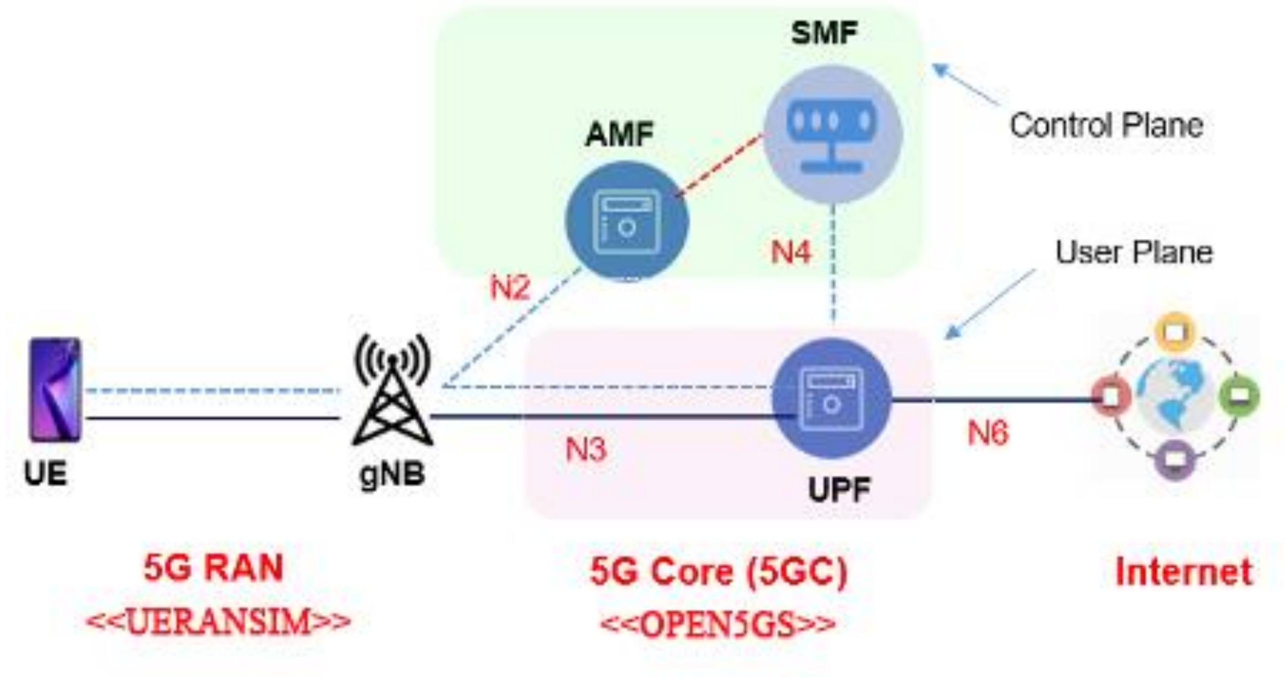
- https://open5gs.org/
- https://github.com/open5gs/open5gs

UERAMSIM

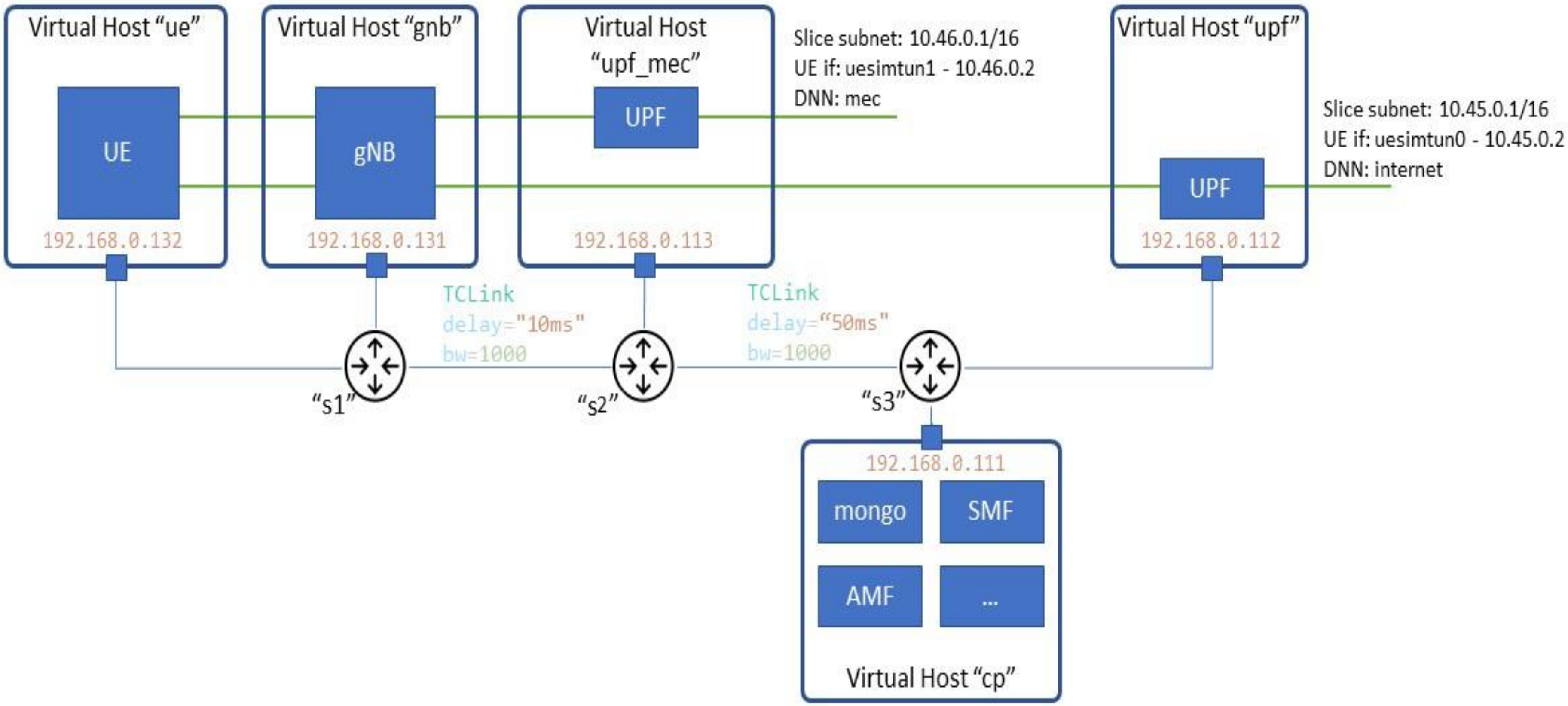
UERANSIM (pronounced "ju-i ræn sɪm"), is the open source state-of-the-art 5G UE and RAN (gNodeB) simulator. UE and RAN can be considered as a 5G mobile phone and a base station in basic terms. The project can be used for testing 5G Core Network and studying 5G System.

1. UERAMSIM
   - GitHub repository https://github.com/aligungr/UERANSIM

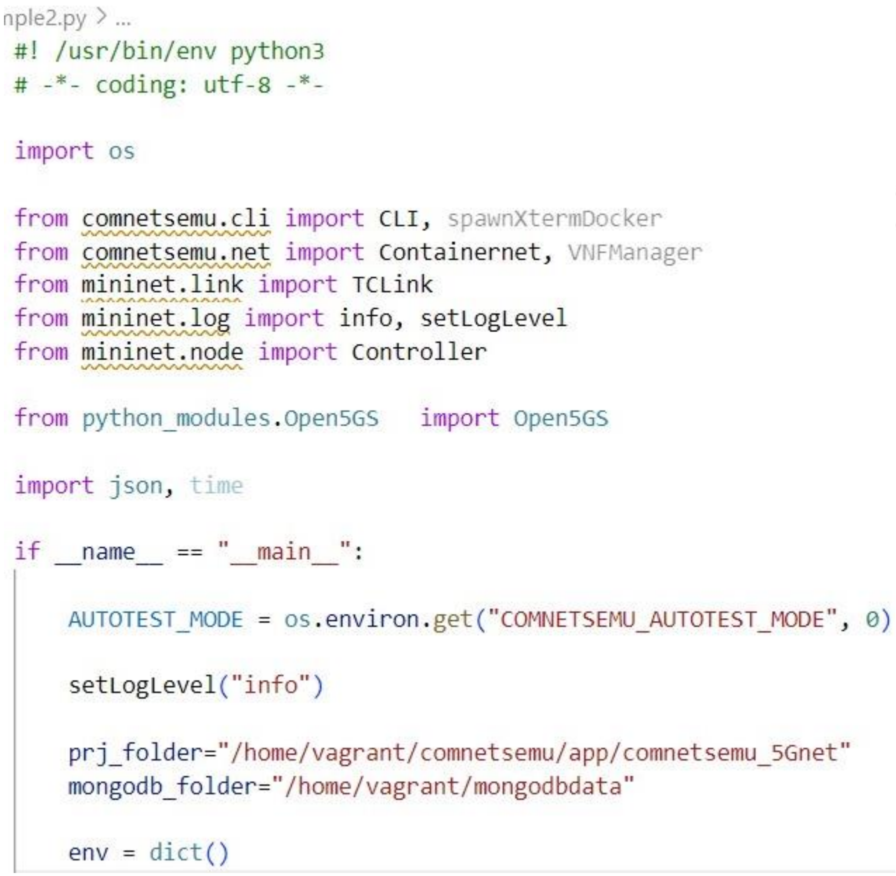
5G network deployment in comnetsemu



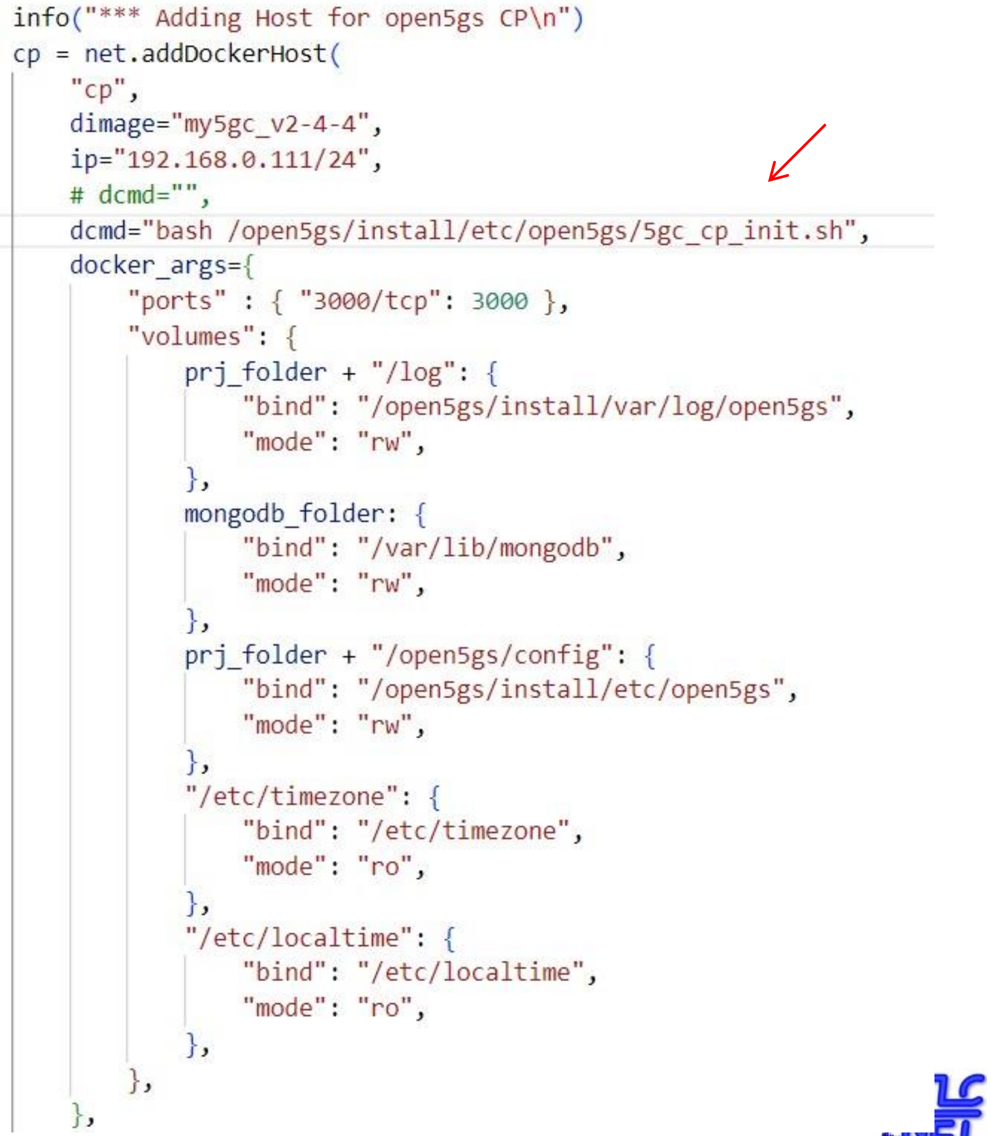
5G network deployment Topology



CP -Topology Implementation (python)

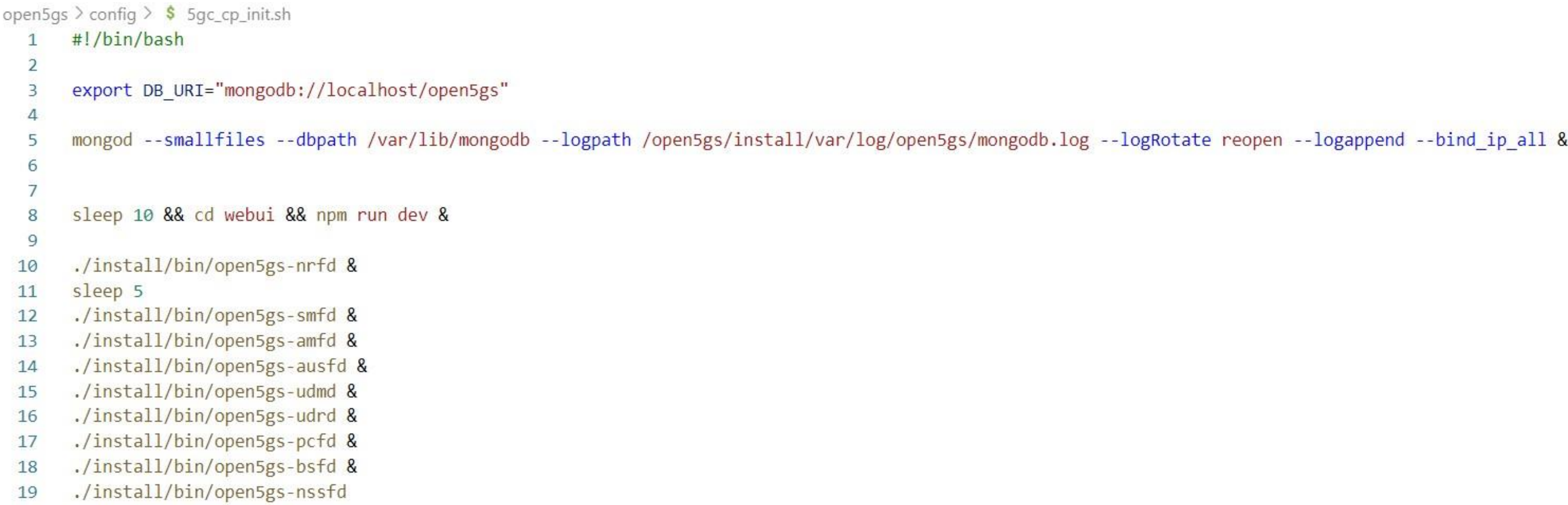


```python
#! /usr/bin/env python3
# -*- coding: utf-8 -*-

import os

from comnetsemu.cli import CLI, spawnXtermDocker
from comnetsemu.net import Containernet, VNFManager
from mininet.link import TCLink
from mininet.log import info, setLogLevel
from mininet.node import Controller

from python_modules.Open5GS   import Open5GS

import json, time

if __name__ == "__main__":

    AUTOTEST_MODE = os.environ.get("COMNETSEMU_AUTOTEST_MODE", 0)

    setLogLevel("info")

    prj_folder="/home/vagrant/comnetsemu/app/comnetsemu_5Gnet"
    mongodb_folder="/home/vagrant/mongodbdata"

    env = dict()
```



```python
info("*** Adding Host for open5gs CP\n")
cp = net.addDockerHost(
    "cp",
    dimage="my5gc_v2-4-4",
    ip="192.168.0.111/24",
    # dcmd="",
    dcmd="bash /open5gs/install/etc/open5gs/5gc_cp_init.sh",
    docker_args={
        "ports" : { "3000/tcp": 3000 },
        "volumes": {
            prj_folder + "/log": {
                "bind": "/open5gs/install/var/log/open5gs",
                "mode": "rw",
            },
            mongodb_folder: {
                "bind": "/var/lib/mongodb",
                "mode": "rw",
            },
            prj_folder + "/open5gs/config": {
                "bind": "/open5gs/install/etc/open5gs",
                "mode": "rw",
            },
            "/etc/timezone": {
                "bind": "/etc/timezone",
                "mode": "ro",
            },
            "/etc/localtime": {
                "bind": "/etc/localtime",
                "mode": "ro",
            },
        },
    },
```

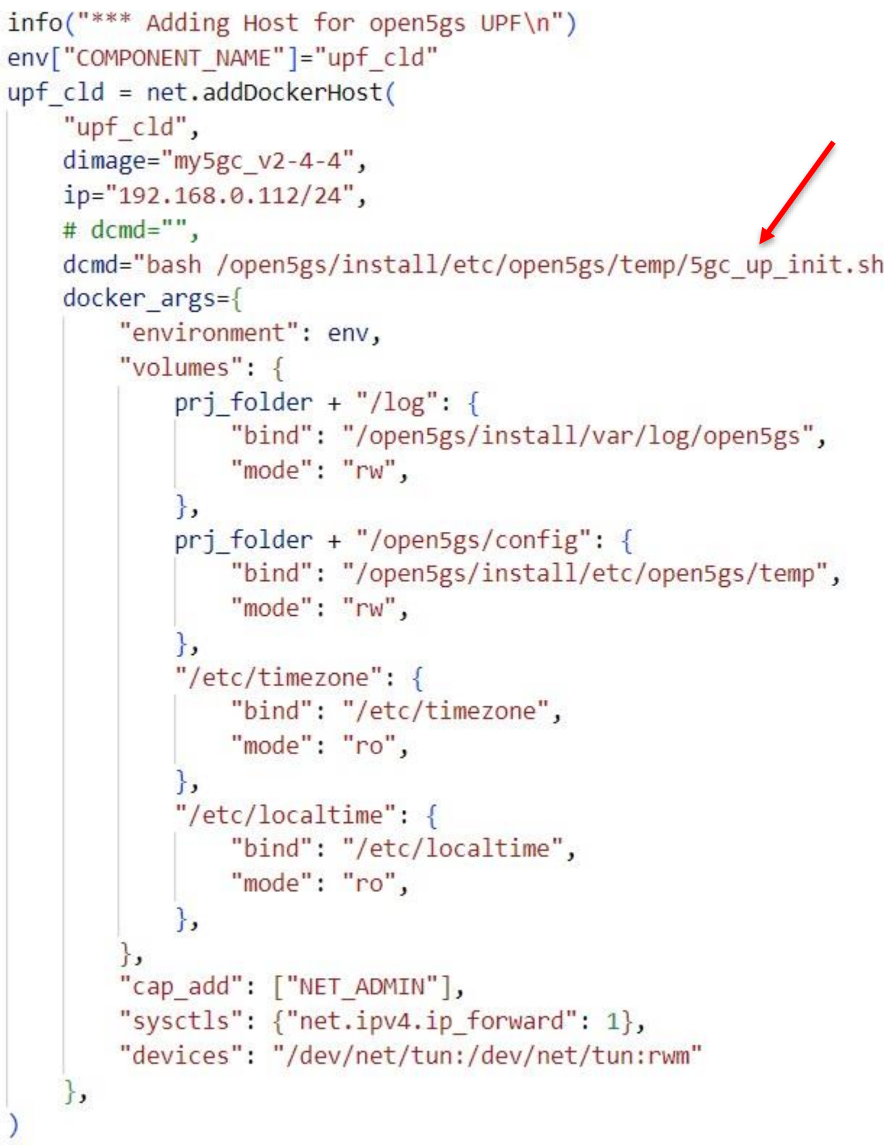
CP - configuration



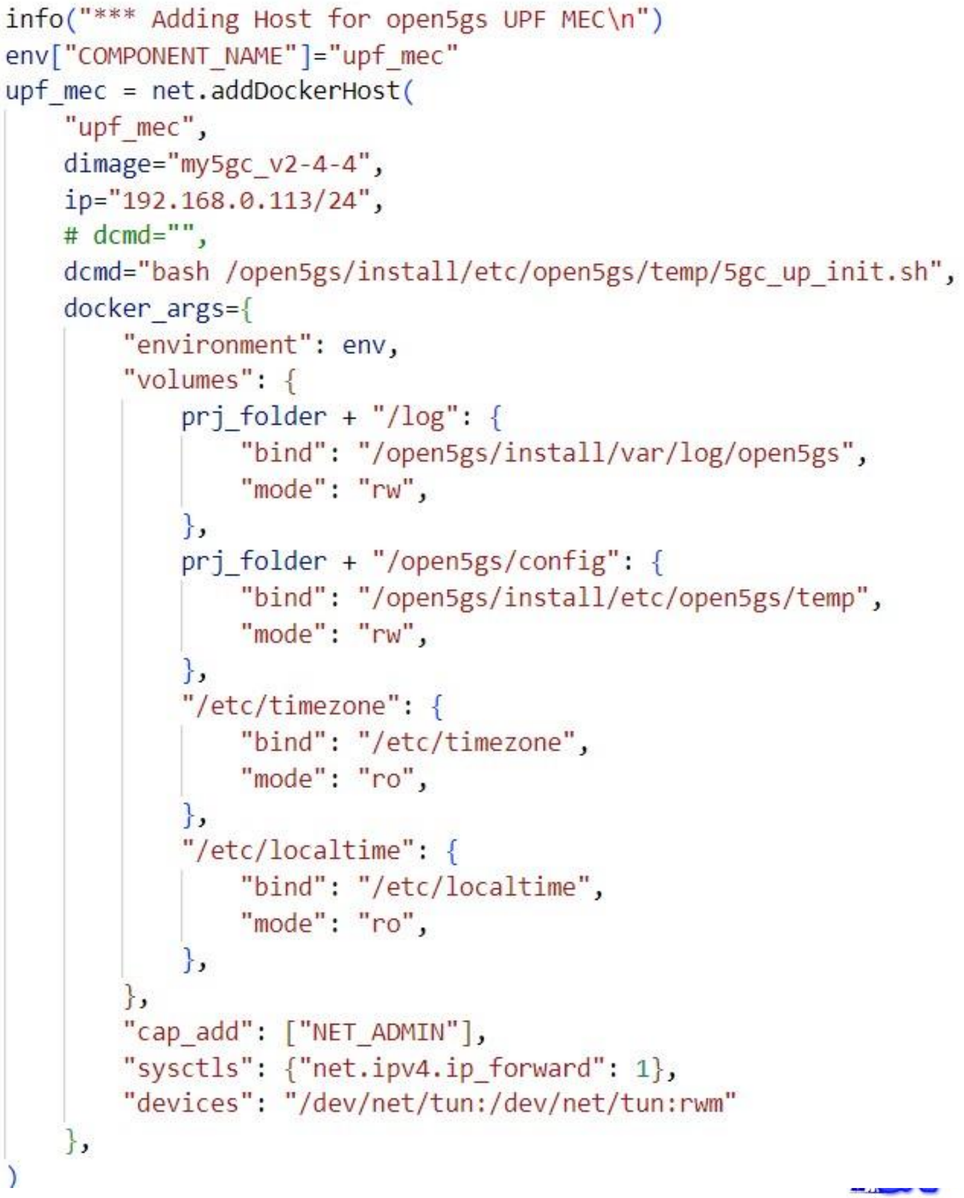
open5gs > config > $ 5gc_cp_init.sh

```bash
#!/bin/bash

export DB_URI="mongodb://localhost/open5gs"

mongod --smallfiles --dbpath /var/lib/mongodb --logpath /open5gs/install/var/log/open5gs/mongodb.log --logRotate reopen --logappend --bind_ip_all &


sleep 10 && cd webui && npm run dev &

./install/bin/open5gs-nrfd &
sleep 5
./install/bin/open5gs-smfd &
./install/bin/open5gs-amfd &
./install/bin/open5gs-ausfd &
./install/bin/open5gs-udmd &
./install/bin/open5gs-udrd &
./install/bin/open5gs-pcfd &
./install/bin/open5gs-bsfd &
./install/bin/open5gs-nssfd
```

UPF-Cloud and UPF-MEC Implementation

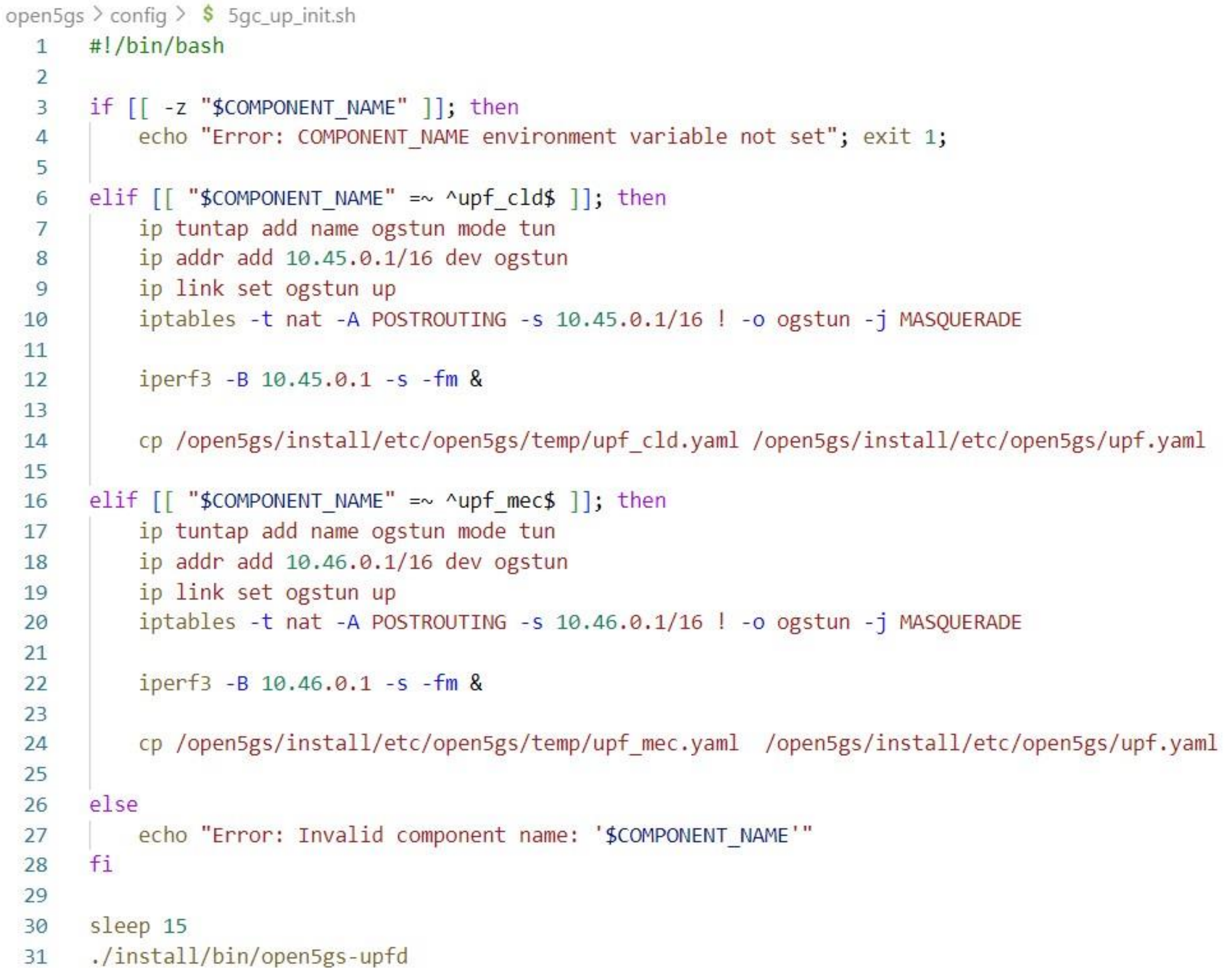


```
info("*** Adding Host for open5gs UPF\n")
env["COMPONENT_NAME"]="upf_cld"
upf_cld = net.addDockerHost(
    "upf_cld",
    dimage="my5gc_v2-4-4",
    ip="192.168.0.112/24",
    # dcmd="",
    dcmd="bash /open5gs/install/etc/open5gs/temp/5gc_up_init.sh
    docker_args={
        "environment": env,
        "volumes": {
            prj_folder + "/log": {
                "bind": "/open5gs/install/var/log/open5gs",
                "mode": "rw",
            },
            prj_folder + "/open5gs/config": {
                "bind": "/open5gs/install/etc/open5gs/temp",
                "mode": "rw",
            },
            "/etc/timezone": {
                "bind": "/etc/timezone",
                "mode": "ro",
            },
            "/etc/localtime": {
                "bind": "/etc/localtime",
                "mode": "ro",
            },
        },
        "cap_add": ["NET_ADMIN"],
        "sysctls": {"net.ipv4.ip_forward": 1},
        "devices": "/dev/net/tun:/dev/net/tun:rwm"
    },
)
```



```
info("*** Adding Host for open5gs UPF MEC\n")
env["COMPONENT_NAME"]="upf_mec"
upf_mec = net.addDockerHost(
    "upf_mec",
    dimage="my5gc_v2-4-4",
    ip="192.168.0.113/24",
    # dcmd="",
    dcmd="bash /open5gs/install/etc/open5gs/temp/5gc_up_init.sh",
    docker_args={
        "environment": env,
        "volumes": {
            prj_folder + "/log": {
                "bind": "/open5gs/install/var/log/open5gs",
                "mode": "rw",
            },
            prj_folder + "/open5gs/config": {
                "bind": "/open5gs/install/etc/open5gs/temp",
                "mode": "rw",
            },
            "/etc/timezone": {
                "bind": "/etc/timezone",
                "mode": "ro",
            },
            "/etc/localtime": {
                "bind": "/etc/localtime",
                "mode": "ro",
            },
        },
        "cap_add": ["NET_ADMIN"],
        "sysctls": {"net.ipv4.ip_forward": 1},
        "devices": "/dev/net/tun:/dev/net/tun:rwm"
    },
)
```

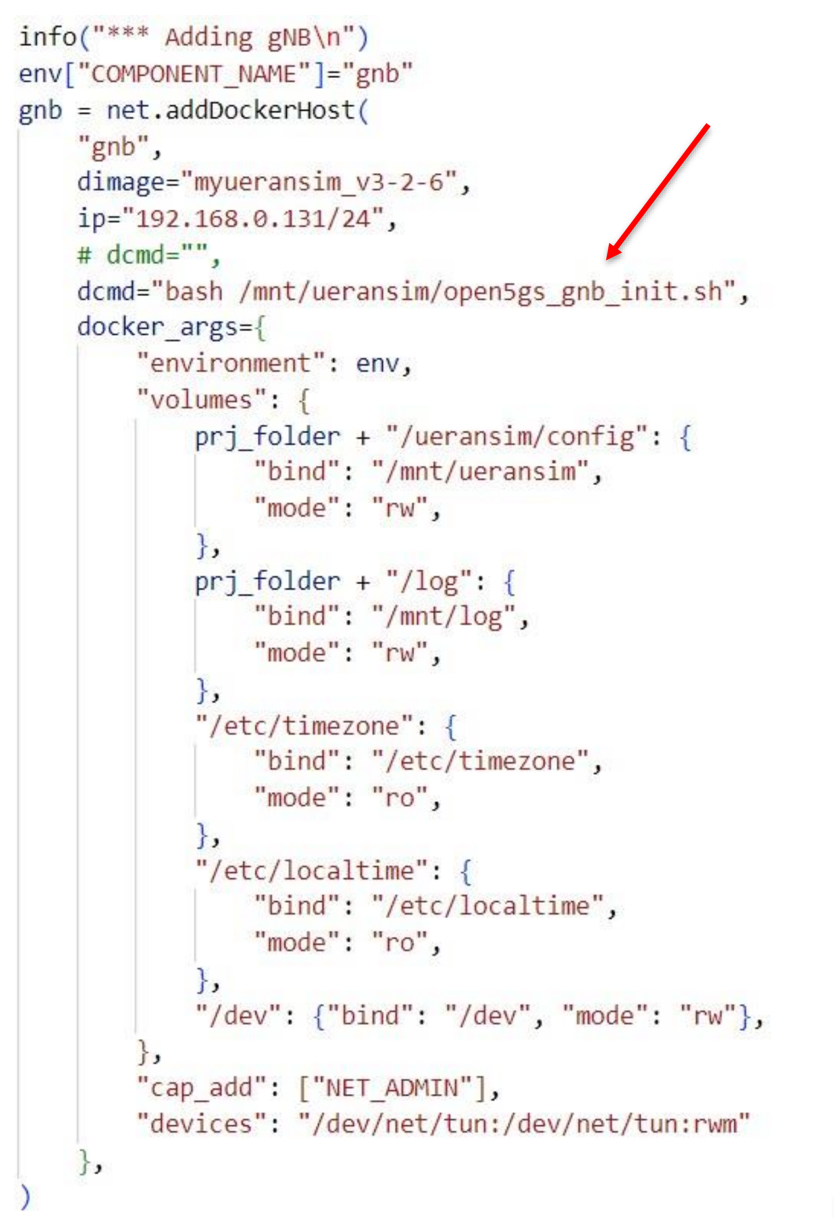
UP – configuration



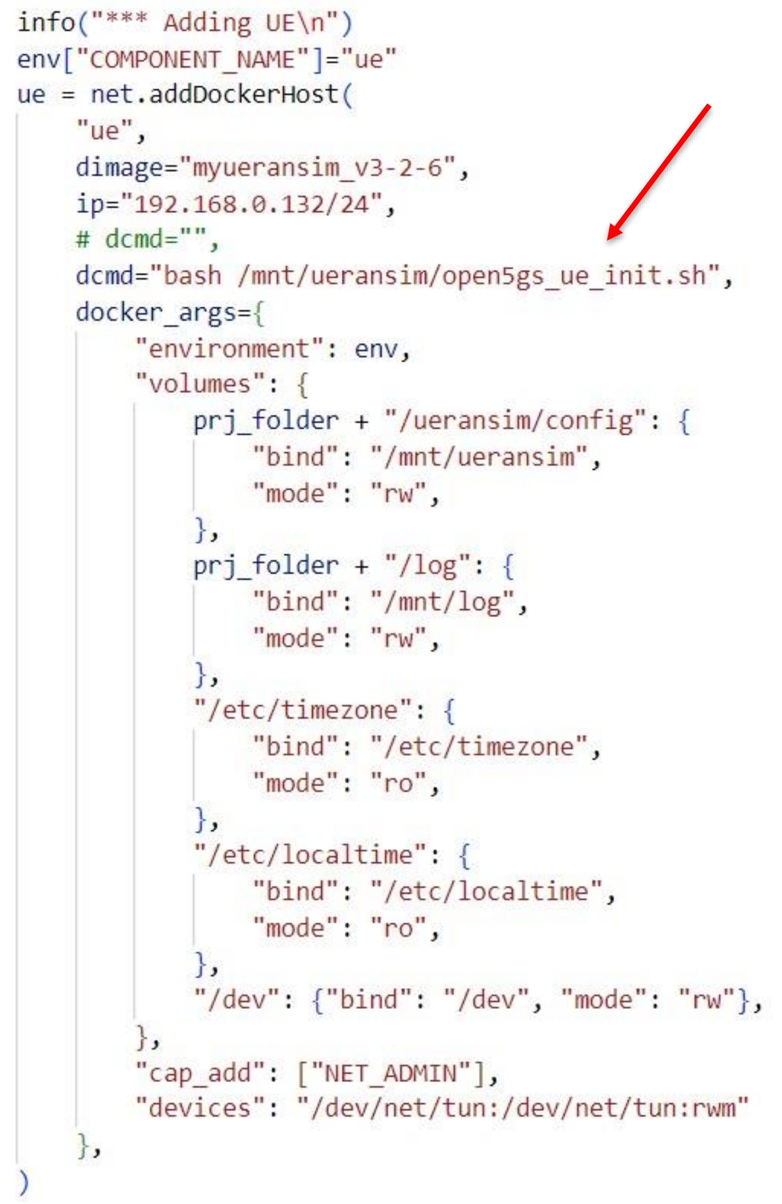
open5gs > config > $ 5gc_up_init.sh

```bash
#!/bin/bash

if [[ -z "$COMPONENT_NAME" ]]; then
    echo "Error: COMPONENT_NAME environment variable not set"; exit 1;

elif [[ "$COMPONENT_NAME" =~ ^upf_cld$ ]]; then
    ip tuntap add name ogstun mode tun
    ip addr add 10.45.0.1/16 dev ogstun
    ip link set ogstun up
    iptables -t nat -A POSTROUTING -s 10.45.0.1/16 ! -o ogstun -j MASQUERADE

    iperf3 -B 10.45.0.1 -s -fm &

    cp /open5gs/install/etc/open5gs/temp/upf_cld.yaml /open5gs/install/etc/open5gs/upf.yaml

elif [[ "$COMPONENT_NAME" =~ ^upf_mec$ ]]; then
    ip tuntap add name ogstun mode tun
    ip addr add 10.46.0.1/16 dev ogstun
    ip link set ogstun up
    iptables -t nat -A POSTROUTING -s 10.46.0.1/16 ! -o ogstun -j MASQUERADE

    iperf3 -B 10.46.0.1 -s -fm &

    cp /open5gs/install/etc/open5gs/temp/upf_mec.yaml  /open5gs/install/etc/open5gs/upf.yaml

else
    echo "Error: Invalid component name: '$COMPONENT_NAME'"
fi

sleep 15
./install/bin/open5gs-upfd
```

GNB and UE Implementation

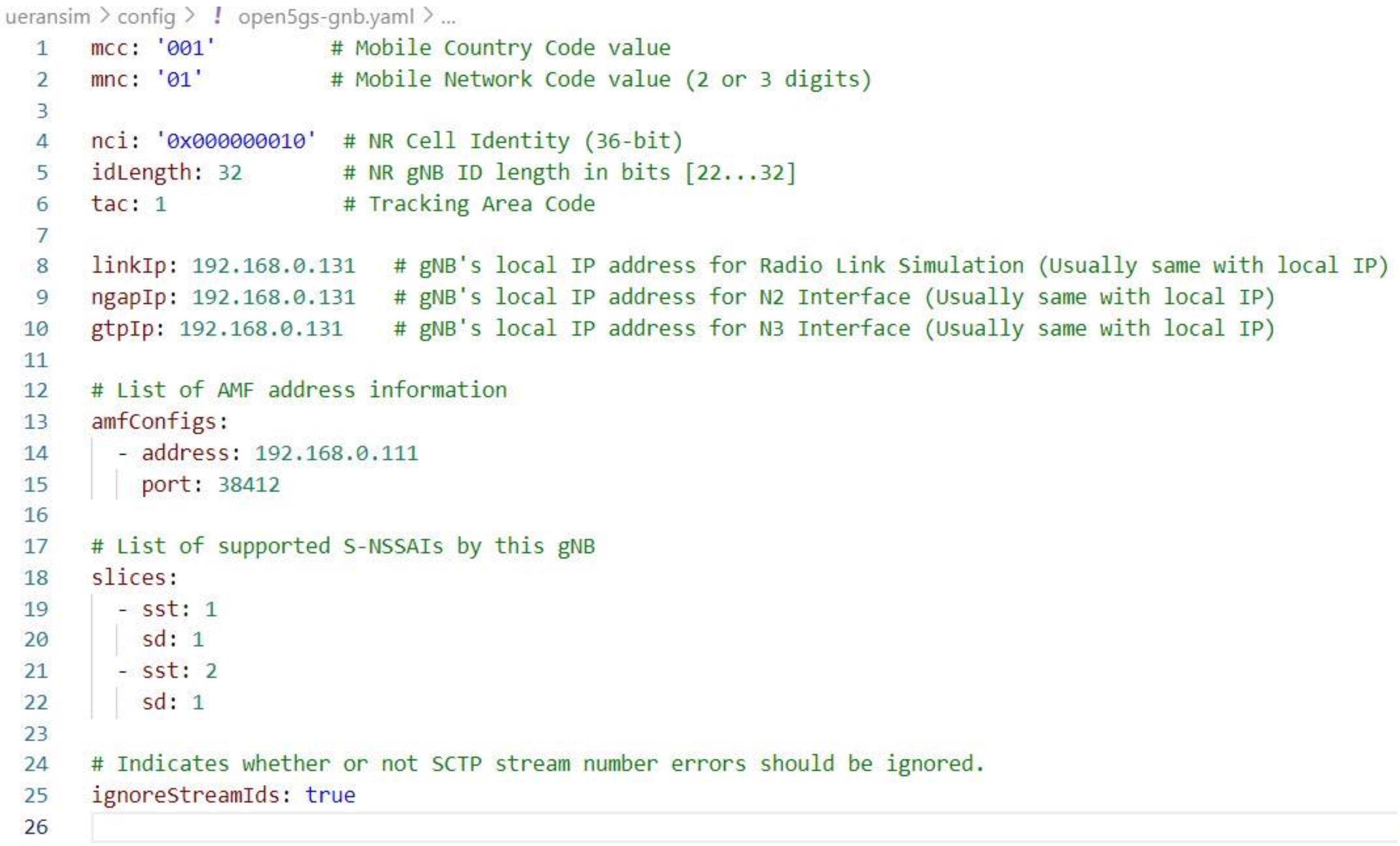


```
info("*** Adding gNB\n")
env["COMPONENT_NAME"]="gnb"
gnb = net.addDockerHost(
    "gnb",
    dimage="myueransim_v3-2-6",
    ip="192.168.0.131/24",
    # dcmd="",
    dcmd="bash /mnt/ueransim/open5gs_gnb_init.sh",
    docker_args={
        "environment": env,
        "volumes": {
            prj_folder + "/ueransim/config": {
                "bind": "/mnt/ueransim",
                "mode": "rw",
            },
            prj_folder + "/log": {
                "bind": "/mnt/log",
                "mode": "rw",
            },
            "/etc/timezone": {
                "bind": "/etc/timezone",
                "mode": "ro",
            },
            "/etc/localtime": {
                "bind": "/etc/localtime",
                "mode": "ro",
            },
            "/dev": {"bind": "/dev", "mode": "rw"},
        },
        "cap_add": ["NET_ADMIN"],
        "devices": "/dev/net/tun:/dev/net/tun:rwm"
    },
)
```



```
info("*** Adding UE\n")
env["COMPONENT_NAME"]="ue"
ue = net.addDockerHost(
    "ue",
    dimage="myueransim_v3-2-6",
    ip="192.168.0.132/24",
    # dcmd="",
    dcmd="bash /mnt/ueransim/open5gs_ue_init.sh",
    docker_args={
        "environment": env,
        "volumes": {
            prj_folder + "/ueransim/config": {
                "bind": "/mnt/ueransim",
                "mode": "rw",
            },
            prj_folder + "/log": {
                "bind": "/mnt/log",
                "mode": "rw",
            },
            "/etc/timezone": {
                "bind": "/etc/timezone",
                "mode": "ro",
            },
            "/etc/localtime": {
                "bind": "/etc/localtime",
                "mode": "ro",
            },
            "/dev": {"bind": "/dev", "mode": "rw"},
        },
        "cap_add": ["NET_ADMIN"],
        "devices": "/dev/net/tun:/dev/net/tun:rwm"
    },
)
```

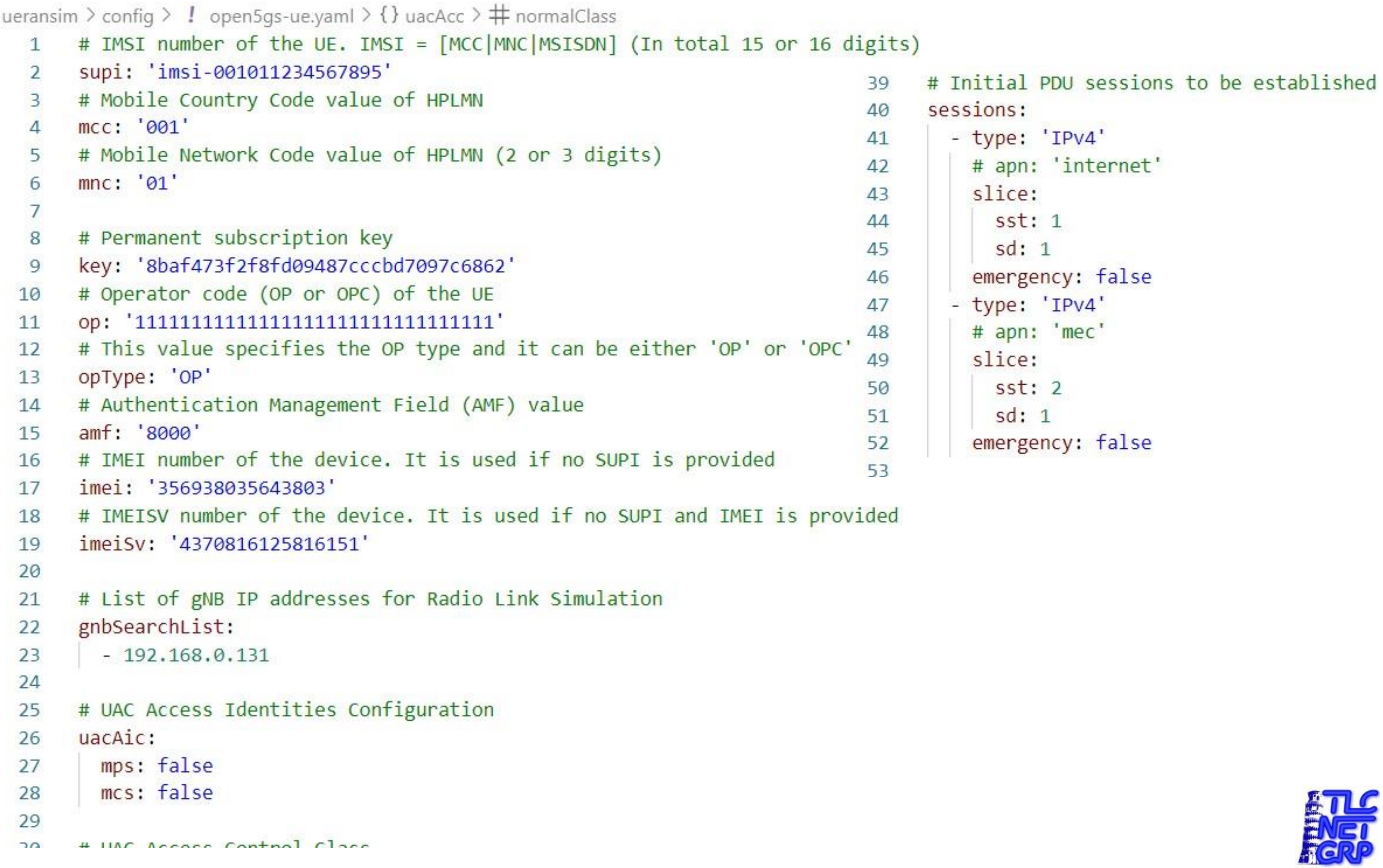
GNB -Configuration



ueransim > config > ! open5gs-gnb.yaml > ...

```yaml
mcc: '001'         # Mobile Country Code value
mnc: '01'          # Mobile Network Code value (2 or 3 digits)

nci: '0x000000010'  # NR Cell Identity (36-bit)
idLength: 32        # NR gNB ID length in bits [22...32]
tac: 1              # Tracking Area Code

linkIp: 192.168.0.131   # gNB's local IP address for Radio Link Simulation (Usually same with local IP)
ngapIp: 192.168.0.131   # gNB's local IP address for N2 Interface (Usually same with local IP)
gtpIp: 192.168.0.131    # gNB's local IP address for N3 Interface (Usually same with local IP)

# List of AMF address information
amfConfigs:
  - address: 192.168.0.111
    port: 38412

# List of supported S-NSSAIs by this gNB
slices:
  - sst: 1
    sd: 1
  - sst: 2
    sd: 1

# Indicates whether or not SCTP stream number errors should be ignored.
ignoreStreamIds: true

```

UE -Configuration

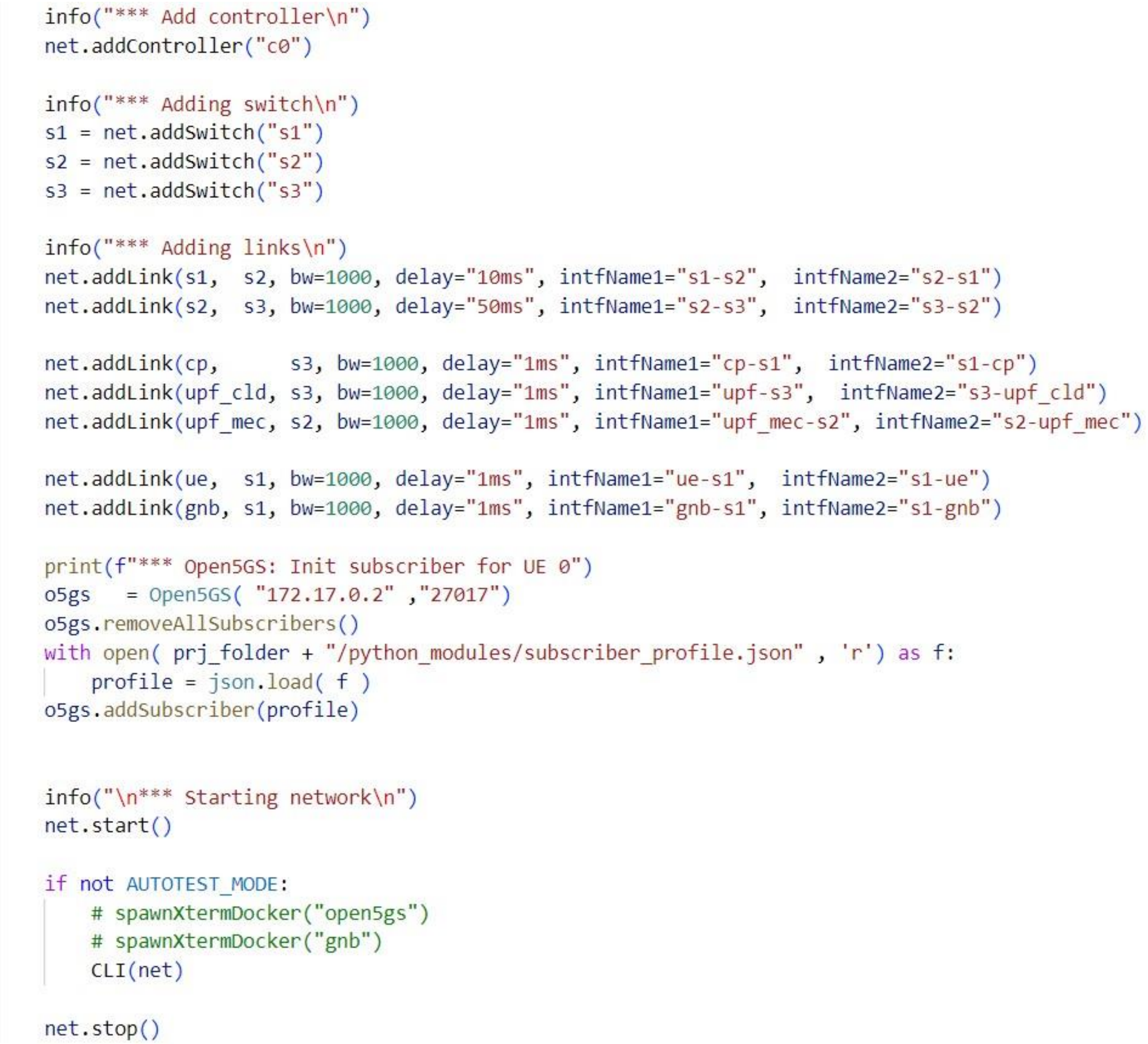


ueransim > config > ! open5gs-ue.yaml > {} uacAcc > # normalClass

```yaml
# IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 or 16 digits)
supi: 'imsi-001011234567895'
# Mobile Country Code value of HPLMN
mcc: '001'
# Mobile Network Code value of HPLMN (2 or 3 digits)
mnc: '01'

# Permanent subscription key
key: '8baf473f2f8fd09487cccbd7097c6862'
# Operator code (OP or OPC) of the UE
op: '11111111111111111111111111111111'
# This value specifies the OP type and it can be either 'OP' or 'OPC'
opType: 'OP'
# Authentication Management Field (AMF) value
amf: '8000'
# IMEI number of the device. It is used if no SUPI is provided
imei: '356938035643803'
# IMEISV number of the device. It is used if no SUPI and IMEI is provided
imeiSv: '4370816125816151'

# List of gNB IP addresses for Radio Link Simulation
gnbSearchList:
  - 192.168.0.131

# UAC Access Identities Configuration
uacAic:
  mps: false
  mcs: false

# UAC Access Control Class
```

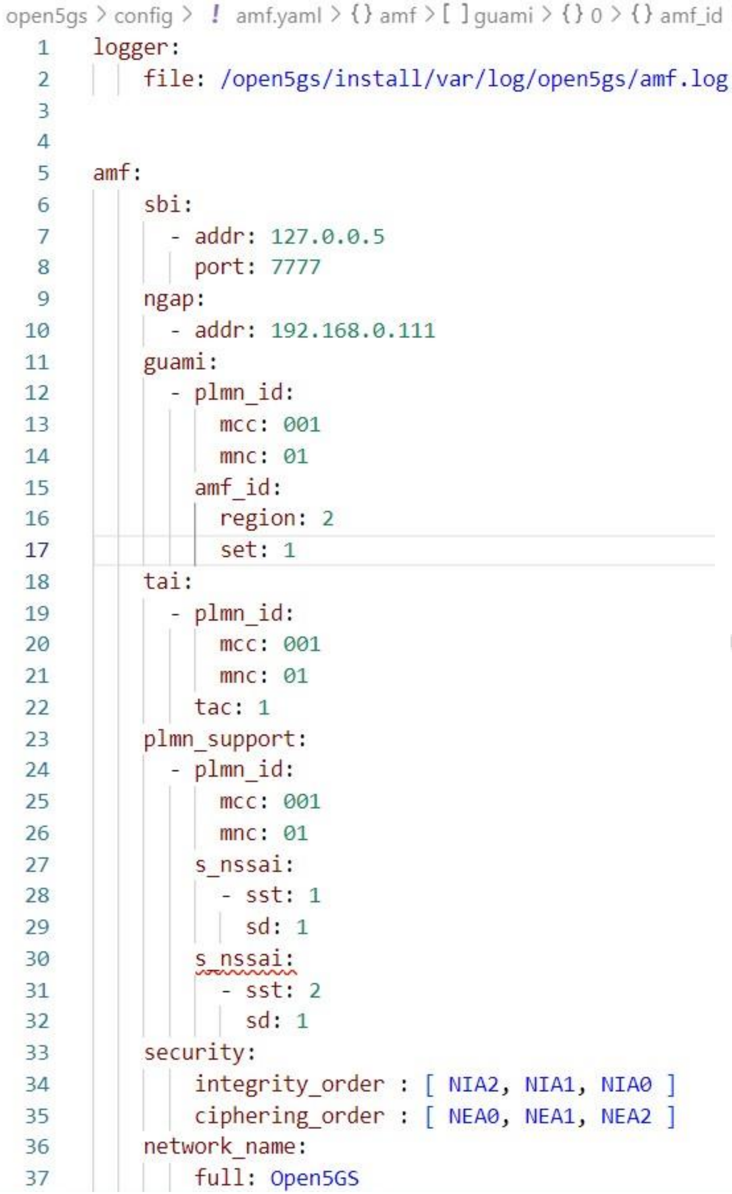
```yaml
# Initial PDU sessions to be established
sessions:
  - type: 'IPv4'
    # apn: 'internet'
    slice:
      sst: 1
      sd: 1
    emergency: false
  - type: 'IPv4'
    # apn: 'mec'
    slice:
      sst: 2
      sd: 1
    emergency: false

```

Networking Implementation



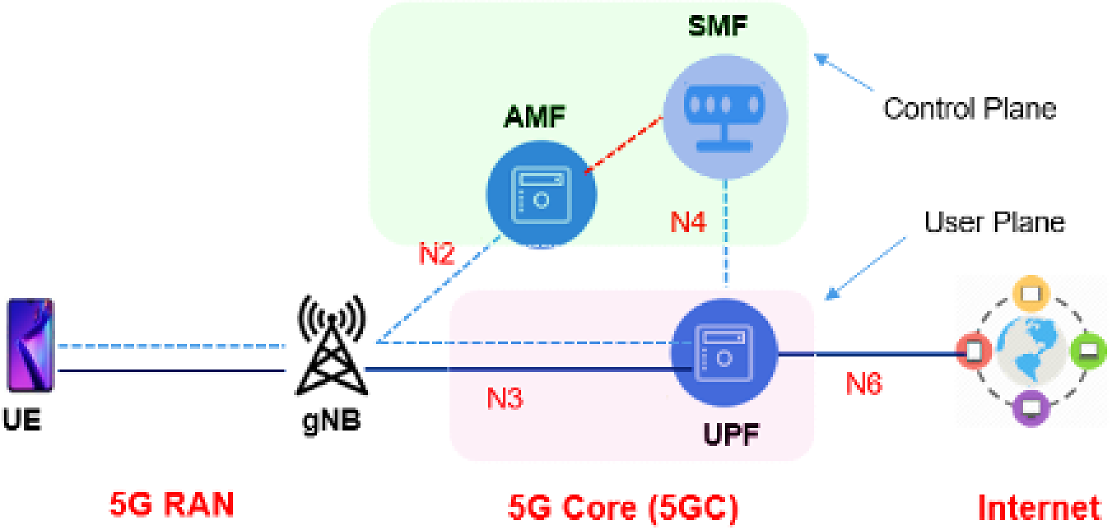
```
info("*** Add controller\n")
net.addController("c0")

info("*** Adding switch\n")
s1 = net.addSwitch("s1")
s2 = net.addSwitch("s2")
s3 = net.addSwitch("s3")

info("*** Adding links\n")
net.addLink(s1,  s2, bw=1000, delay="10ms", intfName1="s1-s2",  intfName2="s2-s1")
net.addLink(s2,  s3, bw=1000, delay="50ms", intfName1="s2-s3",  intfName2="s3-s2")

net.addLink(cp,      s3, bw=1000, delay="1ms", intfName1="cp-s1",  intfName2="s1-cp")
net.addLink(upf_cld, s3, bw=1000, delay="1ms", intfName1="upf-s3",  intfName2="s3-upf_cld")
net.addLink(upf_mec, s2, bw=1000, delay="1ms", intfName1="upf_mec-s2", intfName2="s2-upf_mec")

net.addLink(ue,  s1, bw=1000, delay="1ms", intfName1="ue-s1",  intfName2="s1-ue")
net.addLink(gnb, s1, bw=1000, delay="1ms", intfName1="gnb-s1", intfName2="s1-gnb")

print(f"*** Open5GS: Init subscriber for UE 0")
o5gs   = Open5GS( "172.17.0.2" ,"27017")
o5gs.removeAllSubscribers()
with open( prj_folder + "/python_modules/subscriber_profile.json" , 'r') as f:
    profile = json.load( f )
o5gs.addSubscriber(profile)


info("\n*** Starting network\n")
net.start()

if not AUTOTEST_MODE:
    # spawnXtermDocker("open5gs")
    # spawnXtermDocker("gnb")
    CLI(net)

net.stop()
```

AMF – configuration

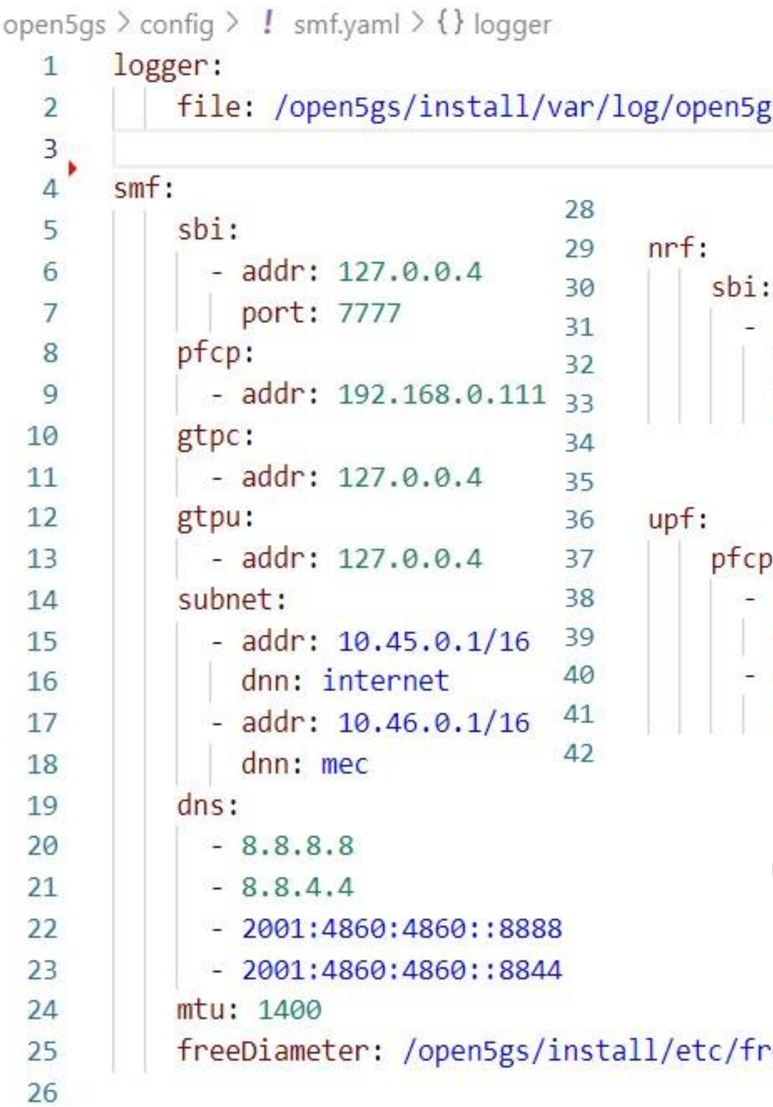


open5gs > config > ! amf.yaml > {} amf > [ ] guami > {} 0 > {} amf_id > # set

```yaml
logger:
    file: /open5gs/install/var/log/open5gs/amf.log


amf:
    sbi:
      - addr: 127.0.0.5
        port: 7777
    ngap:
      - addr: 192.168.0.111
    guami:
      - plmn_id:
          mcc: 001
          mnc: 01
        amf_id:
          region: 2
          set: 1
    tai:
      - plmn_id:
          mcc: 001
          mnc: 01
        tac: 1
    plmn_support:
      - plmn_id:
          mcc: 001
          mnc: 01
        s_nssai:
          - sst: 1
            sd: 1
        s_nssai:
          - sst: 2
            sd: 1
    security:
        integrity_order : [ NIA2, NIA1, NIA0 ]
        ciphering_order : [ NEA0, NEA1, NEA2 ]
    network_name:
        full: Open5GS
```



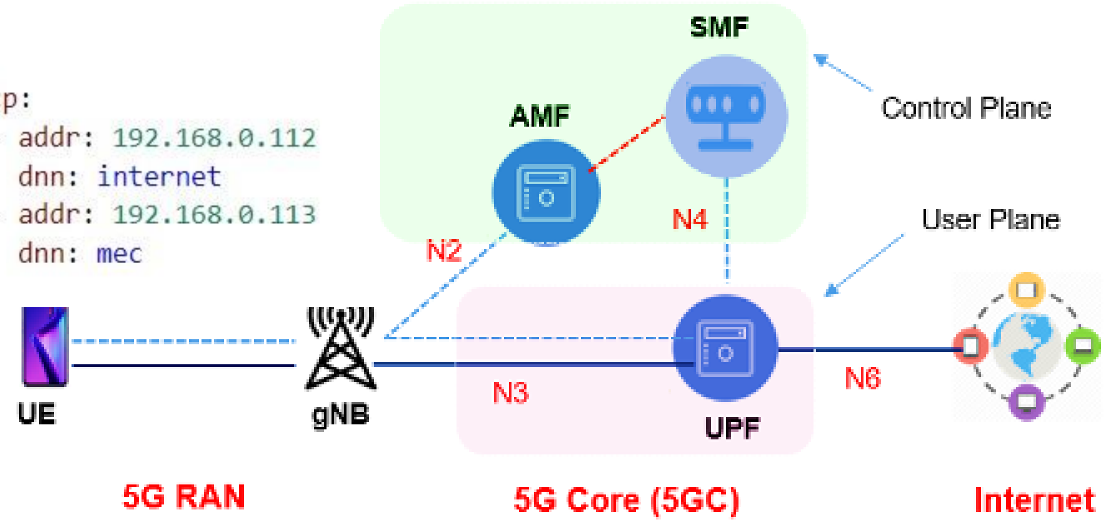
SMF and UPF – configuration



open5gs > config > ! smf.yaml > {} logger

```yaml
logger:
    file: /open5gs/install/var/log/open5gs/smf.log

smf:
    sbi:
      - addr: 127.0.0.4
        port: 7777
    pfcp:
      - addr: 192.168.0.111
    gtpc:
      - addr: 127.0.0.4
    gtpu:
      - addr: 127.0.0.4
    subnet:
      - addr: 10.45.0.1/16
        dnn: internet
      - addr: 10.46.0.1/16
        dnn: mec
    dns:
      - 8.8.8.8
      - 8.8.4.4
      - 2001:4860:4860::8888
      - 2001:4860:4860::8844
    mtu: 1400
    freeDiameter: /open5gs/install/etc/freeDiameter/smf.conf

```

```yaml

nrf:
    sbi:
      - addr:
          - 127.0.0.10
        port: 7777


upf:
    pfcp:
      - addr: 192.168.0.112
        dnn: internet
      - addr: 192.168.0.113
        dnn: mec

```



LeTs
IMPLeMeNT